Text Vectorization

→ One hot Encoding

	Text	Output
D1	The food is good	1
D2	The food is bad	0
D3	Pizza is amazing	1

Vocabulary:	The,	food	is	good	bad	pizza	amazing
D1	1	0	0	0	0	0	0
	0	1	0	0	0	0	0
	⋮	⋮	⋮	⋮	⋮	⋮	⋮

D1 [[1000000],[0100000],[0010000],[0001000]] 4x7

D2 [[1000000],[0100000],[0010000],[0000100]] 4x7

D3 [[0000010],[0010000],[0000001]] 3x7

> Advantages

① Easy to implement

> Disadvantages

① Sparse matrix → overfitting

② Not fixed size input

③ No semantic meanings

④ Out of vocabulary (OOV)

CamScanner

→ Bag of words

Text	Output
He is a good boy	1
She is a good girl	1
Boy and girl are good	1

⇓ lowercase, stopwords

he good boy	1
good girl	1
boy girl good	1

⟹

~~Vocabul~~

Vocabulary	frequency
good	3
boy	2
girl	2

	good	boy	girl
S1	1	1	0
S2	1	0	1
S3	1	1	1

> Advantages

1. Simple and Intutive
2. Fixed size input

> Disadvantages

1. Sparse matrix
2. Ordering of the word changes
3. Out of vocabulary
4. Semantic meanings not captured

→ N-grams

An n-gram is a sequence of n adjacent symbols such as words, letters, syllables or phonemes

Types:

- unigram
- bi-gram
- trigram

		food	not	good
S1 → The food is good	→	1	0	1
S2 → The food is not good	→	1	1	1

> Unigrams

S1 → The, food, is, good

S2 → The, food, is, not, good

> Bi-gram

S1 → The food, food is, is good

S2 → The food, food is, is not, not good

> Tri-gram

S1 → The food is, food is good

S2 → The food is, food is not, is not good.

> Unigram, Bigram Vectorization

	food	not	good	food good	food not	not good
S1	1	0	1	1	0	0
S2	1	1	1	0	1	1

→ TF-IDF [Term Frequency - Inverse Document Frequency]

S1 → good boy

S2 → good girl

S3 → boy girl good

$$\text{Term Frequency} = \frac{\text{No. of rep of words in sentence}}{\text{No. of words in sentence}}$$

$$\text{IDF} = \log_e\left(\frac{\text{No. of sentences}}{\text{No. of sentences contain word}}\right)$$

Term Frequency

	S1	S2	S3
good	1/2	1/2	1/3
boy	1/2	0	1/3
girl	0	1/2	1/3

IDF

words	IDF
good	$\log_e(3/3) = 0$
boy	$\log_e(3/2) = 0.17$
girl	$\log_e(3/2) = 0.17$

TF-IDF

	good	boy	girl
S1	(1/2)(0)	1/2 × 0.17	0
S2	0	0	1/2(0.17)
S3	0	1/3(0.17)	1/3(0.17)

> Advantages

① Intuitive

② Fixed size vocabulary

③ Word importance is captured

> Disadvantages

① Sparsity still exist

② Out of Vocabulary

→ Word Embeddings

In natural language processing (NLP), word embedding is a term used for the representation of words for text analysis, typically in the form of a real-valued vector that encodes the meaning of the words that are closer in the vector space are expected to be similar in meanings

Angry → Vectors

Happy → Vectors

Excited → Vectors

Word Embeddings

- Count or Frequency
  - → One hot encoding
  - → Bag of words
  - → TF-IDF
- Deep Learning
  - → Word2Vec
    - CBOW
    - Skip-gram

→ Word2Vec

Word2Vec is a technique for NLP published in 2013. The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggested additional words for a partial sentence. As the name applies, Word2vec represents each distinct word with a particular list of numbers called a vector.

Vocabulary → Unique words → Corpus

Feature Representation

	Boy	girl	King	queen	apple	mango
Gender	-1	1	-0.92	0.92	0.01	0.23
Royal	0.01	0.02	0.95	0.96	-0.02	0.02
Age	0.03	0.02	0.75	0.68	0.95	0.96
Food	–	–	–	–	0.93	0.91
⋮	–	–	–	–	–	–
n^{th}	–	–	–	–	–	–

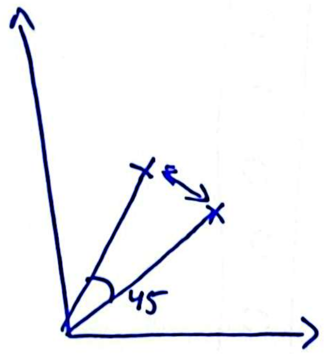
KING – MAN + QUEEN = WOMAN

>Cosine Similarity

KING [0.95, 0.96] MAN [0.95, 0.98]

QUEEN [-0.96, 0.95] WOMAN [-0.94, -0.96]

KING - MAN + QUEEN = WOMAN



Distance = 1 - Cosine Similarity

Cosine Similarity = $\cos\theta$

$= \cos 45 = \frac{1}{\sqrt{2}} = 0.7071$

Distance = 1 − 0.7071
= 0.29

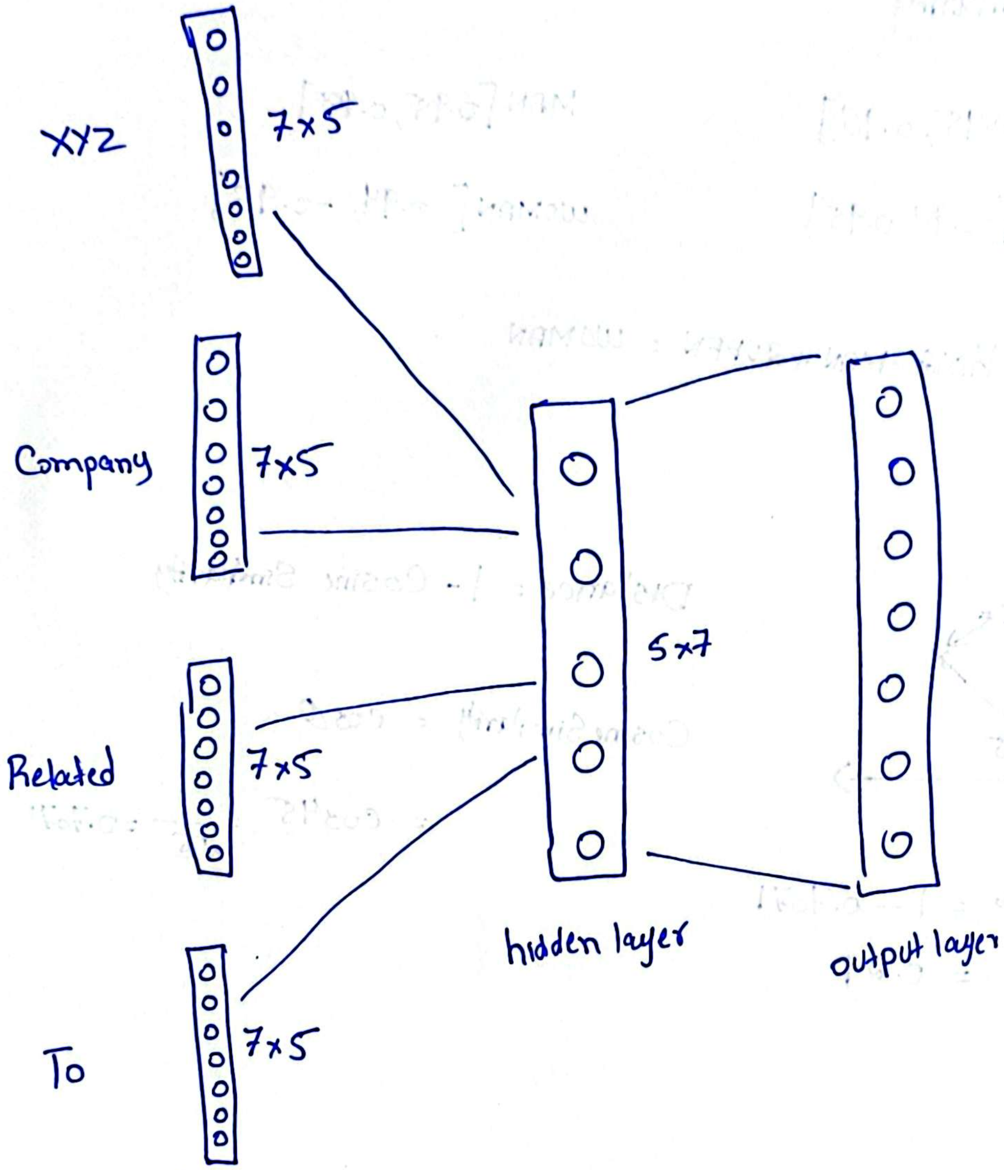
>CBOW

[XYZ Company is related to Data Science]

window_size = ? , let say 5

XYZ Company is related to Data Science

Input	Output
→ XYZ, Company, related, to	is
→ Company, is, to, Data	related
→ is, related, Data, Science	to

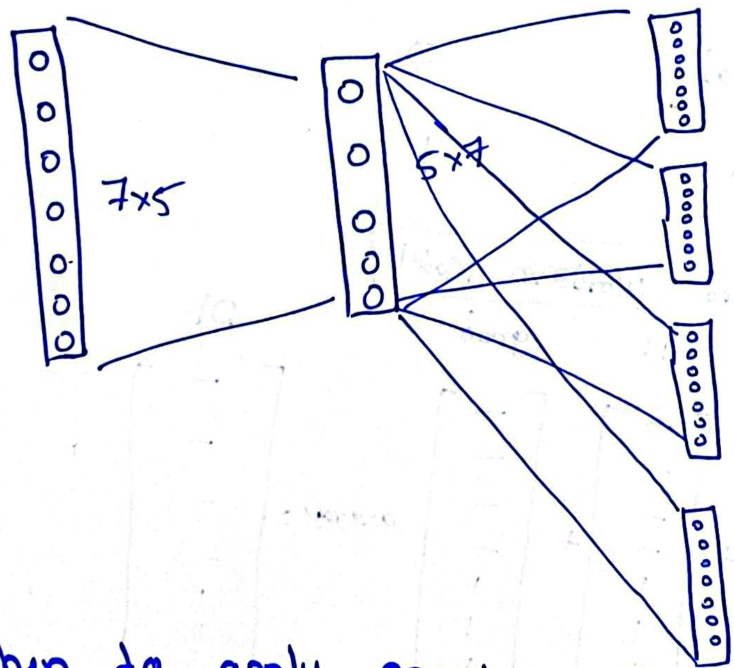

XYZ
7x5
Company
7x5
Related
7x5
To
7x5
5x7
hidden layer
output layer

>Skip-gram

XYZ Company is related to data science.

Window size = 5

Input	output
is	XYZ, Company, related, to
related	Company, is, to, data
to	is, related, data, science



> When to apply CBOW or Skipgram

Small dataset ⟶ CBOW

Huge dataset ⟶ Skipgram

> How to Improve CBOW or skipgram

① Increase Training data

② Increase window size

> Advantages of Word2Vec

① Sparse matrix ⟶ Dense matrix

② Semantic information is captured

③ Vocabulary size ⟶ fixed size of dimension

④ Out of Vocabulary is solved

⟶ Average Word2Vec

	Text	Output
D1	The food is good	1
D2	The food is bad	0
D3	Pizza is amazing	1

$\boxed{\text{Google pretrained Word2Vec Model}}$

$$\begin{array}{ccccccccc} \text{The} & & \text{food} & & \text{is} & & \text{good} & & \text{D1} \\ \begin{bmatrix} - \\ - \\ - \\ - \\ \vdots \end{bmatrix} & + & \begin{bmatrix} - \\ - \\ - \\ - \\ \vdots \end{bmatrix} & + & \begin{bmatrix} - \\ - \\ - \\ - \\ \vdots \end{bmatrix} & + & \begin{bmatrix} - \\ - \\ - \\ - \\ \vdots \end{bmatrix} & \text{average} = & \begin{bmatrix} - \\ - \\ - \\ - \\ \vdots \end{bmatrix} \end{array}$$

300-dimensions